Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development

Sascha Nägele, Nathalie Schenk, Nico Fechtner and Florian Matthes
School of Computation, Information and Technology, Technical University of Munich, Germany

Keywords: Large-Scale Agile Development, Security, Governance, Compliance.

Abstract: Companies are increasingly adopting agile methods at scale, revealing a challenge in balancing team autonomy and organizational control. To address this challenge, we propose an adaptive approach for security governance in large-scale agile software development, based on design science research and expert interviews. In total, we carried out 28 interviews with 18 experts from 15 companies. Our resulting approach includes a generic organizational setup of security-related roles, a team autonomy assessment model, and an adaptive collaboration model. The model assigns activities to roles and determines their frequency based on team autonomy, balancing the autonomy-control tension while ensuring compliance. Although framework-agnostic, we applied our approach to existing scaling agile frameworks to demonstrate its applicability. Our evaluation indicates that the approach addresses a significant problem area and provides valuable guidance for incorporating security into scaled agile environments. While the primary focus is on security governance, our insights may be transferable to other cross-cutting concerns.

1 INTRODUCTION

Driven by the success of agile software development (ASD) methodologies, the application of agile approaches at scale has seen a marked increase (Uludağ et al., 2022). Scaling such development efforts is complex, particularly when there are stringent quality requirements to be met (Kalenda et al., 2018). This complexity raises the issue of balancing team autonomy and decentralized decision-making against the necessity for organizational control and alignment in large-scale agile development (LSAD) (Nägele et al., 2022).

Concurrently, security, a paramount information systems and software quality requirement, is growing in significance due to increasing threats and resulting regulatory requirements (Tayaksi et al., 2022; Moyón et al., 2021). This trend intensifies the tension between autonomy and control, elevating security governance and compliance to a critical concern in LSAD (Moyón et al., 2021).

Existing research indicates that established scaling agile frameworks neither adequately incorporate security activities nor provide sufficient guidance on security and security compliance (Edison et al., 2021; Nägele et al., 2023). For instance, Dännart et al. (2019) observe that security is frequently treated as an isolated concern, and there is a deficiency in "knowing where to conduct which security activity in a large scale agile process" (p. 531). Moyón et al. (2021) suggest that adapting scaling frameworks, e.g., by introducing additional roles, may be beneficial. In addition, based on a systematic literature review and expert interview study, Nägele, Schenk, and Matthes (2023) propose employing influencing factors such as product risk and team maturity to tailor governance and control mechanisms, thereby alleviating the identified tension and enabling increased autonomy and responsibility for more proficient teams. Despite these advances, guidance on how to establish a suitable balance between autonomy and control is lacking.

To fill this research gap, we propose an adaptive approach that aims to enable agile team autonomy while maintaining effective development control without imposing unnecessary burdens. Our primary goal is to establish a balance between agility and (security) governance, resulting in a clear secure software development process that is systematic, transparent, and auditable, while ensuring compatibility with agile environments at scale. We aim to achieve this integration by emphasizing collaboration, continuous improvement, and team autonomy.

Our research question (RQ) is: *How can secu-*

Nägele, S., Schenk, N., Fechtner, N. and Matthes, F.
Balancing Autonomy and Control: An Adaptive Approach for Security Governance in Large-Scale Agile Development.
DOI: 10.5220/0012605000003690
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 26th International Conference on Enterprise Information Systems (ICEIS 2024) - Volume 2*, pages 17-28
ISBN: 978-989-758-692-7; ISSN: 2184-4992
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

rity governance be integrated into LSAD while ensuring adaptability to achieve a suitable equilibrium between autonomy and control?

In the pursuit of answering our RQ, we developed an adaptive approach using design science research (DSR) and expert interviews.

The approach systematically adapts autonomy by considering key influencing factors such as product risk and the capability of agile development teams, granting higher autonomy to more mature teams. It also proposes an organizational setup to foster security collaboration in agile environments at scale.

The paper is structured as follows. Section 2 covers background and related work. Section 3 explains our research method. Section 4 introduces our adaptive approach for security governance in LSAD, evaluated in Section 5. Sections 6 and 7 discuss our findings and conclude our research.

2 BACKGROUND AND RELATED WORK

The following section summarizes the most important definitions as well as the existing challenges and solutions that are relevant to our RQ.

2.1 Security Governance and Compliance

When using the term *governance*, we mainly refer to the domain of *information security governance*, which encompasses the organizational structure and processes to ensure information security. Information security aims to safeguard "the integrity of information, continuity of services and protection of information assets" assets" (Williams, 2001, p. 64). The closely related term *security compliance* can be defined as "the state of conformance with [...] imposed functional security requirements and of providing evidence (assurance) thereof" thereof" (Julisch, 2008, p. 72). The origin of these requirements could be either internal or external (such as required by regulatory bodies, industry standards, or clients).

2.2 Challenges of Security in ASD

To fulfill these internal and external requirements, organizations frequently resort to non-agile security development life cycles, incorporating linear governance and compliance processes within ASD methodologies (Rindell et al., 2018).

This dichotomy presents significant challenges, as widely acknowledged in the current literature. For instance, a systematic mapping study by Moyon et al. (2020) reveals that ensuring security compliance often poses challenges for organizations employing ASD methods, as the required effort grows with shorter development and delivery cycle times. Activities such as integrating security experts and assessors or documenting security compliance evidence can present major hurdles.

Furthermore, Bartsch (2011) identified three main types of challenges for integrating security in agile development, based on literature: challenges related to process aspects, communication and interaction, and trust in individuals and teams.

Trust is closely related to the concept of autonomous teams in ASD. When speaking of *team autonomy*, we refer to "self-organizing teams" or "self-managing teams", as they are synonymously described and defined by researchers in the area of autonomous agile teams (Stray et al., 2018).

2.3 Challenges of Security in LSAD

The tension between agility and security is amplified when trying to scale agile processes while maintaining security compliance without compromising flexibility and speed (Moyón et al., 2021). In regards to scaling agile processes, Dikert et al. (2016) reviewed multiple definitions of LSAD and propose to define *large-scale agile development* as a "software organization with 50 or more people or at least six teams" (Dikert et al., 2016, p. 88).

While the majority of the literature focuses on the general friction between agile methodologies and security, a few authors already investigated security challenges particularly in LSAD.

Nägele, Schenk, and Matthes (2023) identified 15 challenges based on a systematic literature review and interview study, partly based on van der Heijden et al. (2018) who previously identified security challenges specific to LSAD environments. One of the key challenges is the prevalent goal of agile teams to pursue team autonomy, while a certain level of control is often still required, especially in regulated environments (Nägele et al., 2023).

Additionally, Edison et al. (2021) identified several security-related challenges when implementing scaling agile frameworks in practice, such as deficient security awareness and the absence of proactive security activities.

We address the identified challenges in the design of our solution artifact as described in Section 4.1.

2.4 Solution Approaches

In addition to addressing known challenges, we draw from existing solution approaches and success factors already described in the existing literature.

Nägele, Schenk, and Matthes (2023) propose five factors to balance team autonomy and organizational control through security governance based on a systematic literature review and interview study. Newton et al. (2019) propose propose twelve critical success factors and related practices to integrate information security within ASD. These include achieving security awareness, security training, early security testing, dedicated security activities, and a straightforward release process considering security. Typical examples of security activities suitable for LSAD environments are threat modeling, security self-assessments, code reviews, or penetration tests (Nägele et al., 2022).

Bartsch (2011) asserts that integrating security and agility could be achieved by promoting a unified understanding of roles and effectively assimilating assurance and documentation activities into the agile process. Bell et al. (2017) conclude that it is possible to develop secure software with agile approaches if, on the one hand, agile teams integrate security activities and are responsible for developing secure software and, on the other hand, receive enough guidance from security experts. It may be helpful for teams to get involved with security if one team member (a designated security expert or a developer) takes over a security role within the team (Bell et al., 2017).

2.5 Similar Approaches

In the course of our research, two publications emerged with partly similar approaches to balance autonomy and control, demonstrating the interest and relevance of the topic.

Petit and Marnewick (2019) propose a release governance approach that is adaptive with regard to their definition of a team's autonomy.

Poth et al. (2020) also also introduce the concept of team maturity and propose that higher maturity should lead to more autonomy.

Our approach aims to offer a more thorough, flexible, and seamlessly integrable solution to the challenges outlined in existing literature. Another key distinguishing feature is our focus on LSAD and security. In addition, we strive for an approach that is rigorous and systematic enough to be applied and successfully audited even in highly regulated environments, while also leveraging the benefits of agile, autonomous teams.

3 RESEARCH METHOD

Our research primarily followed a DSR approach which facilitates the creation and evaluation of artifacts that address an "unsolved problem or [...] a known problem in a more effective or efficient manner" (Hevner et al., 2004, p.81). Peffers et al. (2007) extended this concept by proposing a concrete DSR process which we use as a blueprint for our research.

We aimed to ensure rigor by building our artifacts on the results of existing literature reviews, e.g., Nägele, Schenk, and Matthes (2023), and additional literature, serving as our knowledge base.

Concurrently, we strove for relevance and practical applicability of our artifacts by employing an interview study, mainly based on Rubin and Rubin (2011) as well as Saldaña (2016), in line with the expert evaluation proposed by Peffers et al. (2012).

In the initial phase of our DSR, we engaged in unstructured interviews and informal discussions with industry partners. Additionally, we drew insights from existing literature as well as our own prior research for problem identification and motivation, as detailed in Sections 1 (Introduction) and 2 (Background and related work) of this paper.

As the second DSR step, based on the identified problem, we derived the objectives of a solution, defined in Section 4.1. We proceeded to design and develop the first iteration of our solution artifacts.

To validate our problem statement, the goals of our solution and the first iteration of our artifacts, we merged the fourth and fifth DSR steps, demonstration and evaluation, by presenting our results to expert interviewees in a first round of interviews and directly collecting feedback. This enabled us to iteratively refine our prototypes, incorporating crucial feedback and producing updated artifact versions.

Subsequently, we conducted a second interview round for the final evaluation. The final solution artifacts are presented in Section 4, with the evaluation results detailed in Section 5. Our key findings and their implications are discussed in Section 6. The communication of our findings through this paper constitutes the final phase of our DSR process.

In total, we carried out 28 interviews with 18 experts from 15 companies across five industries: finance, retail and e-commerce, consulting, entertainment, and automotive. 10 of the experts were interviewed twice and 8 experts were interviewed once, as it was not possible to interview all the experts twice due to time and capacity constraints. The average interview duration was 52 minutes, excluding initial introductions, instructions, and a final wrap-up.

We aimed for a diverse participant pool, encompassing ASD roles as well as security governance and audit-related positions. Specific roles that were included are Product Owner, Scrum Master, Agile Developers, Security Engineer, Security Architect, Solution Architect, Business Analyst, Security Consultant, (IT) Management Consultants, Information Security Lead, Security Officer, Security Manager, and (Security) Auditors. During selection, we prioritized experts with more extensive self-reported experience at the intersection of LSAD and security.

With the interviewees' consent, interviews were recorded and transcribed to ensure accuracy. We adhered to a cyclic approach as endorsed by Salda˜ña (2016), initially coding the data in the first cycle and filtering it more precisely in the second cycle to concentrate on the most pertinent aspects. To streamline coding, we employed the software *MAXQDA*, which facilitates efficient data processing and storage.

4 ADAPTIVE APPROACH FOR SECURITY GOVERNANCE IN LSAD

In this section, we present the resulting artifacts of our research. Those are (i) a generic organizational structure of security-related roles and (ii) a team autonomy assessment model, both integrated within (iii) an adaptive collaboration model, completed by (iv) a projection of the model onto the most relevant scaling agile frameworks. Figure 1 shows an overview of the results and the structure of this section.

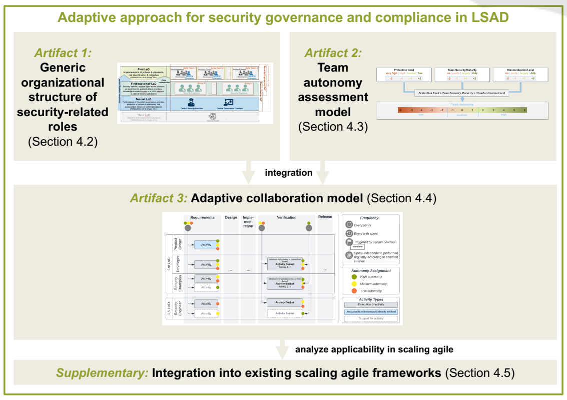

Adaptive approach for security governance and compliance in LSAD
Artifact 1: Generic organizational structure of security-related roles (Section 4.2)
Artifact 2: Team autonomy assessment model (Section 4.3)
integration
Artifact 3: Adaptive collaboration model (Section 4.4)
analyze applicability in scaling agile
Supplementary: Integration into existing scaling agile frameworks (Section 4.5)

Figure 1: Overview of resulting artifacts.

Together, the resulting models form an approach for security governance in LSAD that integrates security-related roles and activities, and adapts to development team capability, as well as organizational and product requirements, thereby balancing autonomy and required control.

4.1 Goals of the Resulting Artifacts

We derived the following six objectives from the research gap and current challenges:

1. ***Achieve a clear definition of roles and security activities without being too prescriptive***, aiming to address the missing guidance on how to conduct security activities in LSAD (van der Heijden et al., 2018; Moyón et al., 2021) and tackle collaboration challenges with non-development functions (Kalenda et al., 2018).
2. ***Attain security compliance without overly rigid governance by balancing the autonomy and control tension***, avoiding controls when not required, and reducing overhead and bottlenecks.
3. ***Enable integration in scaling agile frameworks*** to increase applicability in practice, since current frameworks lack guidance on security and the control autonomy tension (Nägele et al., 2023).
4. ***Increase the security awareness and expertise of agile teams*** to address the previously reported lack thereof (Moyón et al., 2021).
5. ***Promote a responsibility shift towards agile teams*** to enable security governance procedures to be more "agile-friendly" and empower autonomous teams while ensuring compliance.
6. ***Facilitate targeted allocation of security resources*** to tackle the general shortage of security practitioners (Moyón et al., 2021).

4.2 Organizational Structure

Figure 2 illustrates the proposed organizational LSAD setup. The structure is guided by the lines of defense (LoD) model, a generic assurance approach adaptable to both security and agile contexts (Wright, 2014).

The primary alteration we introduce is the "first-and-a-half LoD", represented by security engineers (SEs). They function in a dual role, actively assisting agile teams in the first LoD while serving as the second LoD by reviewing other teams. To avoid conflicts of interest or self-examination, SEs should only assess teams and products not directly within their supportive purview. The designation as engineers emphasizes their software and security engineering expertise, enabling them to support several teams in performing security activities and increasing automation.

Following Wright's proposal, agile teams form the first LoD (Wright, 2014) and are generically composed of a product owner (PO) and developers. For simplicity, we have excluded the scrum master from our model, but their involvement is not precluded.

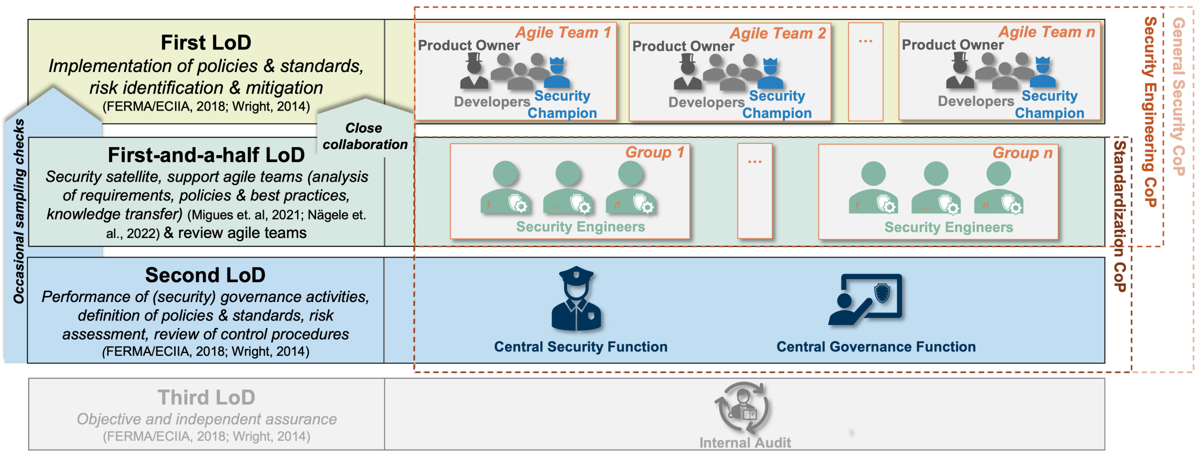

First LoD
Implementation of policies & standards, risk identification & mitigation
(FERMA/ECIIA, 2018; Wright, 2014)
Close collaboration
First-and-a-half LoD
Security satellite, support agile teams (analysis of requirements, policies & best practices, knowledge transfer) (Migues et. al, 2021; Nägele et. al., 2022) & review agile teams
Second LoD
Performance of (security) governance activities, definition of policies & standards, risk assessment, review of control procedures
(FERMA/ECIIA, 2018; Wright, 2014)
Third LoD
Objective and independent assurance
(FERMA/ECIIA, 2018; Wright, 2014)
Occasional sampling checks
Agile Team 1
Agile Team 2
Agile Team n
Product Owner
Developers
Security Champion
Group 1
Group n
Security Engineers
Central Security Function
Central Governance Function
Internal Audit
Standardization CoP
Security Engineering CoP
General Security CoP

Figure 2: Generic organizational structure of security-related roles.

Moreover, we introduce the security champion (SC), a role envisioned not as a new entity but as a developer within the agile team possessing an additional commitment to security. The SC serves as the team's primary security contact point, both within the team and for external stakeholders. The roles of SEs and SCs are not entirely novel, as researchers such as Nägele et al. (2022) have already delineated common responsibilities of these roles from literature and observations in practice.

The second LoD establishes policies and standards, and confirms that the first LoD delivers acceptable work in compliance with these guidelines. In our model, the second LoD also has an enabling role, providing security tools, automated CI/CD pipelines, and training. Unlike the first and first-and-a-half LoDs, the second LoD does not directly participate in the individual teams' development activities.

The third LoD is charged with providing an independent assessment of the work performed by the first and second LoDs. Internal audits, being independent of the development process and typically conducted at random or set intervals, do not necessitate regular interaction with other LoDs in the context of our model.

4.3 Team Autonomy Assessment Model

To balance the required control and granted autonomy, we propose to systematically assess a team autonomy score based on three influencing factors: protection need, team security maturity, and standardization level.

4.3.1 Protection Need

The *protection need* is essential given the distinct risk profiles of software products. These necessitate varying types and scopes of security measures, as well as differing levels of rigor in the validation and verification processes, ultimately impacting the degree of team autonomy that can be granted.

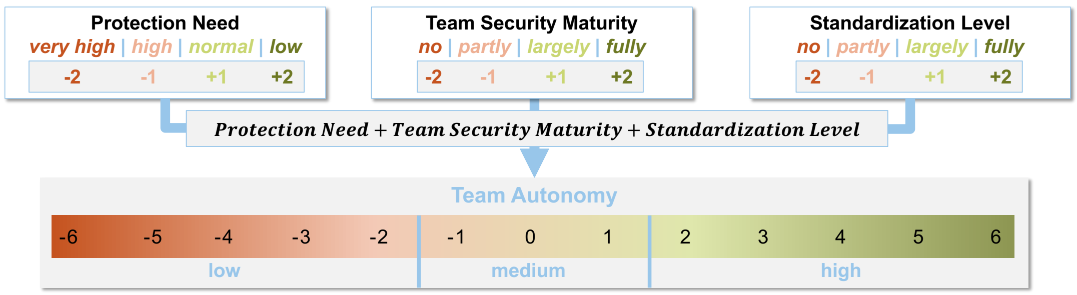
The protection need may encompass the individual product risk and additional factors, including the organization's risk appetite and regulatory requirements. We propose that the protection need assessment is conducted by the PO, who bears product responsibility, in collaboration with an SE representing the second LoD. The SE contributes specialized knowledge to evaluate security dimensions accurately, whereas the PO ensures the consideration of the business perspective.

Our model introduces four protection need levels, omitting a middle option to preclude selection without thorough deliberation. Organizations may employ common risk analysis and predefined damage scenarios to ascertain the appropriate protection need level for a developed product. The more severe the impact of these scenarios, the higher the corresponding protection need level. Examples include non-compliance with laws or regulations, operational incapacitation, and financial harm. The configuration of damage scenarios and their impact severity per protection need level is highly organization-specific. Thus, we refrain from a more detailed definition within the model.

4.3.2 Team Security Maturity

The *team security maturity* reflects a team's capability to develop secure and regulatory-compliant software. We incorporate this factor based on the intuitive supposition that higher maturity levels warrant greater autonomy, as supported in the literature (Poth et al., 2020; Petit and Marnewick, 2019).

The comprehensive design of a maturity model to assess such a capability can be found in one of our previously published studies (Nägele et al.,



Protection Need
very high | high | normal | low
-2 -1 +1 +2
Team Security Maturity
no | partly | largely | fully
-2 -1 +1 +2
Standardization Level
no | partly | largely | fully
-2 -1 +1 +2
Protection Need + Team Security Maturity + Standardization Level
Team Autonomy
-6 -5 -4 -3 -2 -1 0 1 2 3 4 5 6
low
medium
high

Figure 3: Team autonomy assessment calculation including exemplary thresholds.

2024). In summary, we advocate for the amalgamation of methods such as self-assessments, evaluations by external stakeholders, and the employment of (semi-)automated metrics to estimate a team's proficiency in pertinent security dimensions. Regular updates to the assessment are advised, with smaller increments to minimize overhead.

4.3.3 Standardization Level

The third determinant of team autonomy is the *standardization level*, which emerged as a prominent factor in our expert interviews.

Our proposed standardization framework comprises three pillars: (i) utilization of standardized components within the software product, (ii) the underlying infrastructure and tooling for development, deployment, and delivery, and (iii) security-related activities.

Reusable components may address cross-team security requirements, such as authentication, authorization, or secret management. Standardized infrastructure facilitates pre-tested and configured environments for teams, exemplified by customizable pipelines for building, testing, and deploying software iterations on trusted platforms. Employing common standards for these pipelines, such as the same static and dynamic security testing tools with organization-optimized rulesets, enhances comparability, quality, and knowledge sharing.

Standardizing security activities through clear guidance on practices like security code reviews or threat modeling enables greater autonomy. The SE's role is crucial in assessing and guiding teams.

Although standardization may be evaluated as part of the team maturity assessment, its significance warrants separate consideration.

4.3.4 Team Autonomy Score

To integrate all three factors into a team autonomy score, our interview findings suggest a preference for a straightforward calculation method to enhance practical applicability. Each factor is assigned one of four proficiency levels determined through prior assessments, with absolute values representing *very negative (-2)*, *negative (-1)*, *positive (+1)*, or *very positive (+2)* influences on team autonomy. The team autonomy score is obtained by summing these values, with thresholds demarcating low (< -1), medium (> -1 *and* ≤ 1), and high (> 1) autonomy.

Figure 3 depicts the logic, scale, and thresholds. This scale enables compensation for unfavorable conditions; for instance, a high protection need, typically necessitating higher control, can be counterbalanced by high team security maturity and extensive standardized component usage, resulting in high autonomy.

This compensation logic aligns with our overarching goal of fostering high team autonomy, removing bottlenecks, and streamlining processes long-term while motivating teams to improve. Without compensation, a high protection need level would automatically yield low autonomy, reducing incentives for security training and maturity improvement. Organizations can tailor the calculation logic to their requirements and context.

4.4 Adaptive Collaboration Model

Our adaptive collaboration model integrates development-related security activities and assigns them to roles defined in the generic organizational setup. It is adaptive by adjusting role assignments and task frequencies based on team autonomy.

Figure 4 depicts the generic structure of the model, featuring key components such as security activities (grouped by phases), security-related roles, and the frequency of activity execution. The model illustrates the impact of team autonomy on the assignment of roles and the frequency of security activities, represented through color-coded markers. The requirement for a team to undertake a marked activity is contingent on its respective level of autonomy. Unmarked activities for a specific level of autonomy

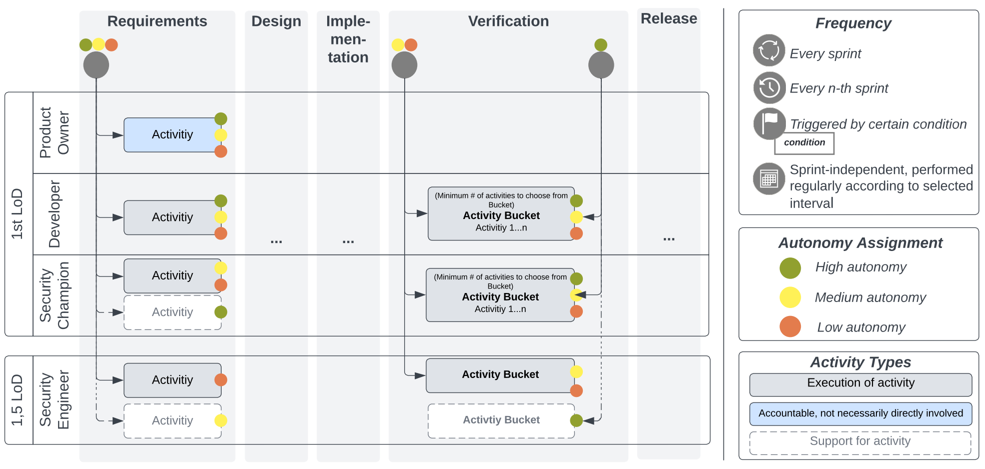


Figure 4: Generic structure of the collaboration model.

are optional and can be disregarded, provided that the team has an adequate autonomy level. However, these assignments should be customized to the specific organization. Generally, we propose that higher autonomy correlates with reduced collaboration with SEs, lower reliance on the SC, and a minimized mandatory frequency of security activities.

To structure and categorize the activities, we derived a common denominator from secure development life cycles in the literature (Boldt et al., 2017; Microsoft, 2012). The model differentiates between *active execution*, *accountability without imposed direct involvement*, and *supporting an activity* (see Figure 4).

It further distinguishes between concrete activities and *activity buckets*, a concept adapted from Microsoft (2012) that enables grouping similar activities. Buckets allow teams to systematically distribute necessary activities over time without having to perform them for every deployment or sprint. Organizations can determine the number of bucket activities to be performed per sprint and establish a cycle to ensure all practices within a bucket are eventually executed.

Concrete examples of security activities are threat modeling, penetration testing, security code and architecture reviews or audits, and implementing improvements according to the findings of static (SAST) and dynamic (DAST) application security testing tools. The selection of specific activities is outside the scope of the model. However, one of our previously published studies provides more in-depth guidance on selecting appropriate security activities in LSAD environments (Nägele et al., 2023).

The model also accommodates custom execution frequencies for individual activities. While some activities might be required every sprint or every n-th sprint, others are only required if a certain trigger is reached, such as a deployment to production. Additionally, certain activities might not be tethered to sprint timelines but rather to designated timespans, e.g., dictated by regulatory requirements.

While the security activities are sequentially categorized, the model should not be interpreted as a linear process. Instead, it serves as an activity catalog, guiding role involvement in necessary security activities. It is essential that team members identify the current category of work to determine the required security measures for that development stage.

4.5 Integration with Scaling Agile Frameworks

This section summarizes the compatibility of our approach with six of the most used scaling agile frameworks in practice (Uludağ et al., 2022): The Scaled Agile Framework (SAFe), Scrum@Scale, Large-Scale Scrum (LeSS), Spotify model, Disciplined Agile (DA) and Nexus.

Overall, our analysis did not discover any irreconcilable conflicts that might prevent the adoption of our model, though certain frameworks necessitate specific considerations.

One potential conflict that might require a softening of framework guidelines is incorporating specialist roles in agile teams. Whereas DA and SAFe are more in line with our recommendation to introduce SCs by describing a specialized role in every team (Ambler and Lines, 2020) or at least the opportunity to include specialists (Knaster and Leffingwell, 2020), other frameworks do not discuss such roles.

In our perspective, these additional roles offer significant value, provided organizations avoid reverting to outdated patterns of specialized roles that do not contribute to development. In our model, the SC is an integral team member, contributing equally to value creation but with an additional 'hat' for security. However, the SC role may be a transitional mechanism until the team achieves sufficient security maturity, at which point it may be redundant. LeSS opposes the introduction of additional roles, advocating instead for 'travelers' – specialists who temporarily join a team to facilitate knowledge transfer before rotating to another (Larman and Vodde, 2016). Organizations using LeSS could deploy SCs as travelers to maintain compatibility.

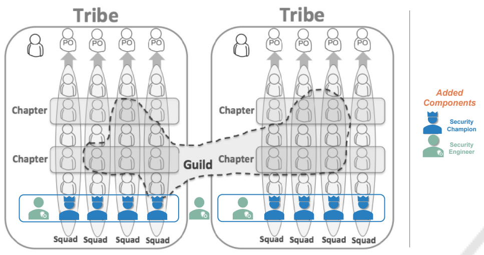


Figure 5: Spotify model exemplary integration.

Introducing team-external roles supporting agile teams is more common and prevalent in all examined frameworks, except for the Spotify model. However, with its concept of chapters and guilds, the Spotify model aligns well with our model, as depicted in Figure 5.

In Nexus, the SE can be included in the integration team, as shown in Figure 6.

The most considerable ambiguity arises concerning the proposed integration of the second LoD. Scrum@Scale includes similar central functions, such as legal and compliance. Other frameworks do not explicitly feature such centralized teams, necessitating a more tailored transfer. Nonetheless, in our model, the second LoD serves as a valuable auxiliary function rather than a core component and could be substituted by other mechanisms. Consequently, we do not perceive this as a significant impediment to the integration of our model.

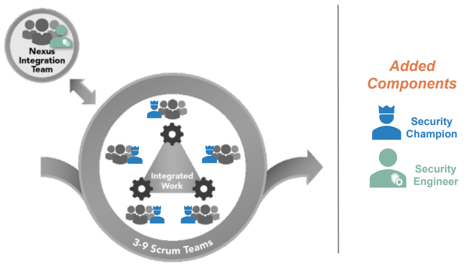


Figure 6: Nexus exemplary integration.

5 EVALUATION

This section summarizes the results of the expert evaluation of our research artifacts. In general, the results are in favor of our approach. Nevertheless, the evaluation also brought interesting controversial aspects to light.

5.1 Organizational Setup Evaluation

In general, all experts considered the proposed organizational setup, including the LoDs, to be clear, rational, and integrable into established systems in their organizations. However, one expert raised concerns about the suitability for smaller organizations due to the additional roles.

The concept of SCs was generally favored, with one exception. Experts suggested role assignment should be affinity-based, not knowledge-based, and proposed SCs act as a central contact for agile developers as well as the second LoD and auditors. Consequently, we expanded the SC's role to include this suggestion. A disputed point was the necessity of an SC or SE in every team. Some experts advocated for a mechanism to assess this necessity, whereas others emphasized only the need for variable allocated capacities. Our model presumes baseline security requirements that necessitate an SC in every team, which most experts agreed with. However, we recognize the mandatory SCs as a limitation that requires further research on its impact on applicability.

The experts stated that successful SC implementation necessitates clearly defined incentives, training roadmaps, and long-term goals. Adequate capacity for their tasks and training is crucial.

Experts agreed on employing SEs to support agile teams and act as reviewers for teams that do not already receive their support. The introduction of the first-and-a-half LoD and its associated SEs received particular praise, especially for ensuring duty separation and in-depth reviews. However, offloading team responsibility to SEs was identified as a potential risk, which our model mitigates by defining SEs as supporters, not primary actors.

The experts confirmed including central governance and security functions in the second LoD to facilitate cross-team collaboration and balance business needs and security requirements. They also supported the translation of generic policies into concrete security standards by central security teams, with SEs assisting in their practical implementation. However, they also discussed alternatives to central teams, such as a security architect role or self-organization through communities, which presuppose a high level of organizational maturity.

Experts agreed with not explicitly including the third LoD, as active involvement would conflict with its audit function. However, they noted interest in the model's promotion of continuous compliance and potential auditing culture shift.

Communities of Practice (CoPs) were generally endorsed for their role in knowledge sharing, although incentives for participation were seen as critical for their success. One expert explained that "enabling knowledge sharing is one of the most important aspects in agile [environments]", thereby stressing the importance of security-focused CoPs.

While deeming our model applicable, one expert suggested a more generic version to account for other types of non-functional requirements.

5.2 Team Autonomy Assessment Evaluation

The experts considered the assessment of team autonomy and influencing factors feasible and valuable, provided that adequate resources such as SEs are available. One expert noted that the assessment aligns well with agile principles and methods, e.g., retrospectives. Another expert voiced concerns about allowing low-autonomy teams to handle high-protection-need projects, suggesting to remove the protection need from the autonomy assessment and assigning the respective projects according to the team autonomy instead. We decided against such a removal but acknowledge the possibility of adding individual constraints or actions to the model, e.g., changing the team-product assignment in case of a "low" assessment result.

An auditor proposed that a maturity score could prove beneficial for tracking the impact and progression of security initiatives over time. However, the expert observed that such scores typically do not suffice to derive suitable measures for enhancing security maturity, potentially leading to overlooked deficiencies, especially when a team's overall maturity rating is high yet has a significant shortfall in a particular area.

No additional factors affecting team autonomy were identified during the evaluation, but a consensus was reached that organizational risk appetite influences protection need and should not be assessed separately. Organizations adopting such an approach should be able to clearly communicate in an audit why certain teams are assumed to have a specific level of security maturity and how the resulting governance approach aligns with the general risk management.

The inclusion of team security maturity was universally valued, particularly when evaluated using objective measurements. While self-assessments were deemed valuable for lean, self-dependent governance, their subjectivity was viewed as a potential drawback.

The standardization level was generally deemed important, despite potential loopholes and workarounds in highly standardized systems. Experts suggested additional aspects, such as the proportion of outsourced components, since these might introduce risks that development teams do not directly control.

5.3 Adaptive Collaboration Model Evaluation

The experts agreed that increased autonomy serves as a compelling incentive for teams to enhance their security focus, thereby facilitating the transition of security responsibilities to the teams. They regarded the model as flexible and adaptable, providing sufficient guidance while preserving room for customization.

One expert emphasized the positive impact of collaborative security activities such as threat modeling and stated that "[...] once you show them how something can go wrong, they are going to be self-motivated to make the thing [the threat] go away. They are going to educate themselves."

Another expert stated that the model's adaptability makes it a "valuable resource-saving approach" by focusing efforts on teams with less autonomy addressing the important lack of security resources issue.

Nonetheless, auditors emphasized the importance of documenting the completion of security activities for compliance, which should be ensured across all teams, irrespective of maturity. Auditors also noted that while the adaptive approach does not conflict with common security standards, initial audits may necessitate closer examination due to its novelty.

The categorization of activities is deemed useful, adding to the model's flexibility. One expert highlighted the inclusion of triggered activities, as it is something the expert has "not seen so far in any other model", but thinks that they provide great value.

We offered illustrative configurations to enhance the model's comprehensibility, encompassing specific security activities. However, the evaluation highlights that the selection and configuration of activities largely depend on the specifics of the organizations.

Overall, the experts were satisfied with the level of complexity, for example stating that "the model is not over-engineered" and that "[...] being too detailed would be the pitfall of the model."

However, while appreciating the model's ability to distill a complex issue, experts expressed concerns about managing the remaining complexity - in particular, conveying the model's configuration to the respective teams considering their team autonomy. We showcased potential solutions for this challenge by using the web-based diagramming tool "Lucidchart" for demonstration. Its simple interactive elements allow to create filters for different maturity levels and a process diagram that adapts to the selected filters, reducing the visualization complexity and enhancing comprehensibility - a factor critical for the model's practical applicability.

5.4 Evaluation of Framework Integration

Expert opinions on incorporating the artifacts into existing scaling agile frameworks varied. Some experts saw merit in offering integration guides, arguing that such guidance could streamline adaptation for companies utilizing these frameworks. Others recognized the potential utility but considered the integration straightforward and thus would not prioritize it. One expert highlighted the possible resistance of agile practitioners to the introduction of specialized roles, stressing the importance of illustrating the congruence between our role model and these frameworks.

6 DISCUSSION

6.1 Key Findings

To answer our RQ and address the identified research gap, we created artifacts that guide in balancing the control and autonomy tension in LSAD. In the following, we present four key findings of our research.

First, the expert evaluation finds that our adaptive approach, anchored in team autonomy assessments, forms a valuable basis for resource-efficient governance for security compliance within LSAD. According to our interviews, the current modus operandi in large-scale enterprises leans towards top-down governance procedures. However, we advocate for a transformation towards autonomy and self-governance to facilitate sustainable security integration into LSAD. Organizations can leverage team autonomy to adapt their governance process and avoid having controls in place when circumstances do not call for them, aiming to ensure security compliance while reducing governance bottlenecks. This approach improves the compatibility of security governance and compliance efforts with agile methods at scale, a key challenge reported in existing research (Nägele et al., 2023).

Second, the experts assessed the collaboration model as valuable. The model provides a customizable distribution of security activities among roles and iterative development phases, thereby offering clear guidance on the collaboration between agile teams and security experts. By incorporating visualization and a filtering functionality, the model enhances its usability and boosts transparency. Team autonomy, which serves as an input to the model, helps to determine the roles that participate in an activity and the frequency of specific activities, thereby achieving a desirable degree of adaptivity. We recommend a reduced collaboration with team-external stakeholders and security specialists for teams with high autonomy, and fewer prescribed security activities and audits. By shifting responsibility to teams, the model aims to conserve scarce security resources while encouraging teams to prioritize security and achieve faster delivery. We thereby tackle challenges described in previous literature, such as the shortage of security practitioners (Moyón et al., 2021) and the need for guidance on integrating security activities in LSAD (Edison et al., 2021; Dännart et al., 2019).

Thirdly, during expert interviews, some intriguing perspectives on the rigidity of our approach emerged. Initially, a few auditors preferred more structure and control, while some agile practitioners found it too rigid and process-based. This divergence in perceptions affirmed the relevance of the tension we aim to reconcile. Through further dialogue, iterations, and the incorporation of the expert feedback into our models, both groups started to appreciate the advantages of our approach. Governance and auditing roles particularly valued the systematic and traceable method for granting autonomy. At the same time, agile practitioners saw value in the approach because it empowers experts to hone and refine their skills, rewarding expertise with greater autonomy and accountability.

Lastly, we acknowledge that our approach is extensive, and its implementation within an organization may require significant effort. However, our approach could also inspire incremental changes to existing established procedures within organizations. Our expert interviews also corroborated this, sparking considerable interest in integrating the perspective of team autonomy within their organizations.

6.2 Limitations

To yield robust results and artifacts, we incorporated expert interviews into the development and final evaluation of our work. Recognizing the potential impact of expert representativeness on the validity of our findings (Miles et al., 2014), we prioritized experts with extensive experience and diverse backgrounds.

Additionally, we acknowledge the potential influence of researchers on the objectivity of the collected data and its analysis (Miles et al., 2014). To mitigate this, we followed the interview questionnaire as closely as possible during the semi-structured interviews and employed cyclic coding in our analysis. Transcription of the interviews further limited the influence of researcher-inherent bias.

Another notable limitation of this study lies in the absence of experiments and practical applications of our approach due to the complexity of the research topic. However, such investigations are currently in progress, even extending to cross-cutting concerns beyond security, such as software architecture and user experience.

7 CONCLUSION

The increasing adoption of LSAD, coupled with the growing significance of security, necessitates approaches to achieve a balance between team autonomy and organizational control. Existing literature recognizes this tension and its resultant challenges, but comprehensive solutions remain scarce. To address this gap, we employed a DSR approach to develop solution artifacts. We combined this approach with an expert interview study to improve our artifacts and ensure practical relevance and applicability. Our approach aspires to harmonize governance with LSAD by using a team autonomy assessment as a determinant of role responsibilities and security activity frequency. Our central recommendation is that more capable and mature teams should receive more autonomy, based on a documented and transparent assessment and process model. This solution balances the granted autonomy and required control while preserving compliance and auditability. The expert evaluations generally endorse our approach while illuminating areas for enhancement and additional research opportunities. Future experiments should scrutinize the practicality of our approach, perhaps focusing initially on a select few teams to increase feasibility before scaling to larger environments. Prospective adaptations of the approach could also encompass other cross-cutting concerns and non-functional requirements beyond security.

ACKNOWLEDGEMENTS

This work has been supported by the German Federal Ministry of Education and Research (BMBF) Software Campus grant 01IS17049.

REFERENCES

Ambler, S. W. and Lines, M. (2020). *Choose Your WoW: A Disciplined Agile Delivery Handbook for Optimizing Your Way of Working*. PM Institute.

Bartsch, S. (2011). Practitioners' perspectives on security in agile development. In *Sixth International Conference on Availability, Reliability and Security (ARES)*, pages 479–484, Piscataway, NJ. IEEE.

Bell, L., Bird, J., Brunton-Spall, M., and Smith, R. (2017). *Agile application security: Enabling security in a continuous delivery pipeline*. O'Reilly Media, Sebastopol, CA.

Boldt, M., Jacobsson, A., Baca, D., and Carlsson, B. (2017). Introducing a novel security-enhanced agile software development process. *International Journal of Secure Software Engineering*, 8(2):26–52.

Dännart, S., Moyón, F., and Beckers, K. (2019). An assessment model for continuous security compliance in large scale agile environments. In *Advanced Information Systems Engineering*, pages 529–544, Cham, Switzerland. Springer.

Dikert, K., Paasivaara, M., and Lassenius, C. (2016). Challenges and success factors for large-scale agile transformations: A systematic literature review. *Journal of Systems and Software*, 119:87–108.

Edison, H., Wang, X., and Conboy, K. (2021). Comparing methods for large-scale agile software development: A systematic literature review. *IEEE Transactions on Software Engineering*, pages 1–23.

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in IS research. *MIS Quarterly*, 28(1), 75–105.

Julisch, K. (2008). Security compliance: The next frontier in security research. *Proc. of the new security paradigms workshop*, 71–74.

Kalenda, M., Hyna, P., and Rossi, B. (2018). Scaling agile in large organizations: Practices, challenges, and success factors. *Journal of Software: Evolution and Process*, 30(10):p. 1954.

Knaster, R. and Leffingwell, D. (2020). *SAFe 5.0 Distilled: Achieving Business Agility with the Scaled Agile Framework*. Addison-Wesley, Boston.

Larman, C. and Vodde, B. (2016). *Large-scale Scrum: More with LeSS*. Addison-Wesley, Boston and Amsterdam and London.

Microsoft (2012). Security development lifecycle - sdl process guidance version 5.2.

Miles, M. B., Huberman, A. M., and Saldaña, J. (2014). *Qualitative data analysis: A methods sourcebook*. SAGE, Thousand Oaks Califorinia, 3 edition.

Moyon, F., Almeida, P., Riofrio, D., Mendez, D., and Kalinowski, M. (2020). Security compliance in agile software development: A systematic mapping study. In *46th Euromicro Conference on Software Engineering* *and Advanced Applications (SEAA)*, pages 413–420. IEEE.

Moyón, F., Méndez, D., Beckers, K., and Klepper, S. (2021). How to integrate security compliance requirements with agile software engineering at scale? In *Product-Focused Software Process Improvement*, pages 69–87, Cham, Switzerland. Springer.

Nägele, S., Korn, L., and Matthes, F. (2023). Adoption of information security practices in large-scale agile software development: A case study in the finance industry. In *Proc. of the 18th Int. Conf. on Availability, Reliability and Security*, ARES '23, New York, NY, USA. Association for Computing Machinery.

Nägele, S., Schenk, N., and Matthes, F. (2023). The current state of security governance and compliance in large-scale agile development: A systematic literature review and interview study. In *IEEE 25th Conference on Business Informatics (CBI)*.

Nägele, S., Watzelt, J.-P., and Matthes, F. (2022). Investigating the current state of security in large-scale agile development. In *23rd Int. Conf. on Agile Software Development (XP)*, pages 203–219. Springer.

Nägele, S., Watzelt, J.-P., and Matthes, F. (2024). Assessing team security maturity in large-scale agile development. In *Proc. of the 57th Hawaii Int. Conf. on System Sciences*, pages 7259–7269.

Newton, N., Anslow, C., and Drechsler, A. (2019). Information security in agile software development projects: A critical success factor perspective. In *27th European Conference on Information Systems (ECIS)*, pages 1 – 17.

Peffers, K., Rothenberger, M., Tuunanen, T., and Vaezi, R. (2012). Design science research evaluation. In *Design Science Research in Information Systems. Advances in Theory and Practice*, volume 7286, pages 398–410. Berlin, Heidelberg.

Peffers, K., Tuunanen, T., Rothenberger, M. A., and Chatterjee, S. (2007). A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77.

Petit, Y. and Marnewick, C. (2019). Earn your wings: A novel approach to deployment governance. In *Agile Processes in Software Engineering and XP – Workshops*, volume 364, pages 64–71, Cham, Switzerland. Springer.

Poth, A., Jacobsen, J., and Riel, A. (2020). Systematic agile development in regulated environments. In *Systems, Software and Services Process Improvement*, volume 1251, pages 191–202, Cham. Springer.

Rindell, K., Hyrynsalmi, S., and Leppänen, V. (2018). Aligning security objectives with agile software development. In *19th Int. Conference on Agile Software Development*, pages 1–9, New York, NY, USA. ACM.

Rubin, H. J. and Rubin, I. S. (2011). *Qualitative Interviewing: The Art of Hearing Data*. SAGE, 3 edition.

Saldaña, J. (2016). *The coding manual for qualitative researchers*. SAGE, Los Angeles and London, 3 edition.

Stray, V., Moe, N. B., and Hoda, R. (2018). Autonomous agile teams: Challenges and future directions for research. In *Proceedings of the 19th International Conference on Agile Software Development: Companion*, pages 1–5, New York, NY, USA. ACM.

Tayaksi, C., Ada, E., Kazancoglu, Y., and Sagnak, M. (2022). The financial impacts of information systems security breaches on publicly traded companies: reactions of different sectors. *Journal of Enterprise Information Management*, 35(2):650–668.

Uludağ, Ö., Philipp, P., Putta, A., Paasivaara, M., Lassenius, C., and Matthes, F. (2022). Revealing the state of the art of large-scale agile development research: A systematic mapping study. *Journal of Systems and Software*, 194.

van der Heijden, A., Broasca, C., and Serebrenik, A. (2018). An empirical perspective on security challenges in large-scale agile software development. In *12th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, pages 1–4, New York, NY. ACM.

Williams, P. (2001). Information security governance. *Information Security Technical Report*, 6(3), 60–70.

Wright, C. (2014). *Agile Governance and Audit*. IT Governance Publishing, Cambridgeshire.